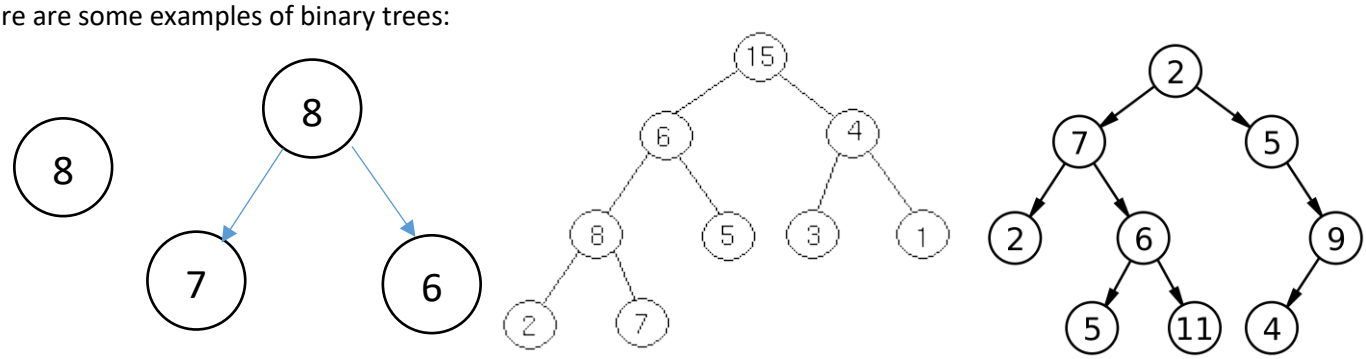
ITCS 209 Object Oriented Programming	Name:	Lab Score	Challenge Bonus
	ID:		

Lab13: Recursion

A binary tree is a tree in which every node has at most two children. Recursively, a full binary tree is either:

1. An empty tree (i.e., NULL)
2. A graph formed by adding a binary tree to the left child and a binary tree to the right child of a non-empty node.

Here are some examples of binary trees:



Hint: More information can be found at:
https://www.tutorialspoint.com/data_structures_algorithms/tree_data_structure.htm

You are given the `Node class that implements a basic node that supports binary tree structure (Do not modify Node.java) and the TreeCalculatorTester` that implements test cases (Do not modify TreeCalculatorTester.java), and `TreeCalculator` class whose methods are left blank for you to fill in. Specifically, you need to implement the following methods:

`public static int` `findMax(Node root)`: Recursively traverse the tree from `root` and return the maximum node `id` in the tree. If the tree pointed by `root` is null, return -1.

`public static int` `findMin(Node root)`: Recursively traverse the tree from the root and return the minimum node `id` in the tree. If the tree pointed by `root` is null, return -1.

You can assume that the valid range of an `id` is [0, Integer.MAX_VALUE-1]. Furthermore, you can implement additional "helper" methods if needed.

Expected output from `testRegular()`:

```
-------------- Regular ----------------
Tree[0] Max: -1      Min: -1
Tree[1] Max: 16      Min: 16
Tree[2] Max: 16      Min: 14
Tree[3] Max: 10      Min: 1
Tree[4] Max: 10      Min: 1
Tree[5] Max: 2147483646    Min: 1
Tree[6] Max: 7       Min: 1
```

Challenge Bonus (Optional):

Implement the following methods.

`public static double sumTree(Node root)`: Return the sum of all nodes. If `root` is null, return 0.

`public static double avgTree(Node root)`: Return the average of all the nodes. If `root` is null, return 0.

Sample output from `testBonus()`:

```
-------------- BONUS ----------------
Tree[0] Sum: 0.0  Average: 0.0
Tree[1] Sum: 16.0 Average: 16.0
Tree[2] Sum: 30.0 Average: 15.0
Tree[3] Sum: 39.0 Average: 5.571428571428571
Tree[4] Sum: 26.0 Average: 5.2
Tree[5] Sum: 2.147483675E9    Average: 3.579139458333333E8
Tree[6] Sum: 17.0 Average: 4.25
```